CHALMERS
UNIVERSITY OF TECHNOLOGY

Finite element method algorithm for geotechnical applications based on Runge-Kutta scheme with automatic error control

Computers and Geotechnics 128 (2020) 103841

Contents lists available at ScienceDirect

Computers and Geotechnics

journal homepage: www.elsevier.com/locate/compgeo

Research Paper

Finite element method algorithm for geotechnical applications based on Runge-Kutta scheme with automatic error control

Ayman A. Abed[a,b,*], Wojciech T. Sołowski[b]

[a] Chalmers University of Technology, Department of Architecture and Civil Engineering, SE-412 96 Göteborg, Sweden
[b] Aalto University, Department of Civil Engineering, P.O. Box 12100, FI-00076 Aalto, Finland

ARTICLE INFO

Keywords:
FEM
Explicit methods
Runge-Kutta scheme
Automatic error control

ABSTRACT

This paper introduces a novel explicit algorithm to solve the finite element equation linking the nodal displacements of the elements with the external forces applied via means of non-linear global stiffness matrix. The proposed method solves the equation using Runge-Kutta scheme with automatic error control. The method allows any Runge-Kutta scheme, with the paper demonstrating the algorithm efficiency for Runge-Kutta schemes of second to fifth order of accuracy. The paper discusses the theoretical background, the implementation steps and validates the proposed algorithm. The numerical tests show that the proposed method is robust and stable. In comparison to the iterative implicit methods (e.g. Newton-Raphson method), the new algorithm overcomes the problem of occasional divergence. Furthermore, considering the computation time, the fifth-order accurate scheme proves to be competitive with the iterative method. It seems that the proposed algorithm could be a powerful alternative to the standard solution procedures for the cases with strong nonlinearity, where the typical algorithms may diverge.

1. Introduction

The Finite Element Method approximates the solution of the continuous mechanical balance equation by calculating the unknown displacement $\mathbf{u}$ at a set of discrete points only, typically at the element nodes, as a response to the external load $\mathbf{f}_{ext}$. This requires solving a large set of equations, with coefficients given in the global stiffness matrix $\mathbf{K}$ where:

$$\mathbf{Ku} - \mathbf{f}_{ext} = 0 \tag{1}$$

In most engineering applications, the stiffness matrix $\mathbf{K}$ is changing in a non-linear fashion with the increment of external load $\mathbf{f}_{ext}$. The stiffness matrix nonlinearity can be a consequence of both the material behaviour and the geometric nonlinearity. In the common geotechnical applications, the material behaviour nonlinearity is dominant. Therefore, this paper disregards geometrical nonlinearity.

To solve Eq. (1), the Finite Element Method needs an efficient, stable and robust numerical algorithm. Typically the method of choice belongs to the family of iterative Newton-Raphson scheme, for example: modified, initial stress or accelerated (Bathe, 2006), which usually offers a satisfactory rate of convergence. Yet, the Newton-Raphson scheme requires the initial iteration result close enough (i.e. within the convergence radius) to the correct solution to converge and an even closer result in order to get the quadratic rate of convergence. On the other hand, if the load increment is large compared to the non-linearity of the problem, the initial iteration result may fall outside of the convergence radius and the Newton-Raphson scheme may fail to reach the solution. If this happens, the codes after significant number of iterations usually cut the load increment to half and repeat the process, hoping that the convergence will be reached. If that does not work, ultimately, the solution may be abandoned with an error (Section 8 gives an example of such situation).

The method also requires the evaluation of the inverse of the Jacobian (first order derivatives) matrix at each iteration which imposes additional numerical expense and difficulties in a large system of equations. To avoid the problems related to the repetitive calculation of the Jacobian, the quasi-Newton methods evolved (e.g. Broyden-Fletcher-Goldfarb-Shanno (BFGS)) (Matthies and Strang, 1979; Avriel, 2003). The main idea behind this class of methods is that the Jacobian matrix in Newton-Raphson scheme needs to be estimated only in the first iteration and then, in the subsequent iterations, it will be approximated based on the Jacobian from the previous iteration. In the original contribution, Brayden (Broyden, 1965) even proposed a formula to update the Jacobian inverse directly based on the inverse of the preceding iteration which would further save the computational effort. Although the quasi-Newton methods offer occasionally successful

* Corresponding author at: Chalmers University of Technology, Department of Architecture and Civil Engineering, SE-412 96 Göteborg, Sweden.
E-mail address: ayman.abed@chalmers.se (A.A. Abed).

https://doi.org/10.1016/j.compgeo.2020.103841
Received 4 March 2020; Received in revised form 10 September 2020; Accepted 12 September 2020
Available online 24 September 2020

Nomenclature

Roman

D_{Tol}	tolerated displacement error
Err	error vector
Error	scalar error measure
$\mathbf{f}_{ext}$	external forces vector, M LT^{-2}
$\mathbf{f}_{int}$	internal nodal forces vector, M LT^{-2}
F	yield function
I_{Tol}	tolerated error for iterative procedure
$\mathbf{K}$	global stiffness matrix
K_o	coefficient of at rest earth pressure
K_o^{NC}	coefficient of at rest earth pressure for normally consolidated soil
m	number of nodes in the mesh
M	slope of critical state line
n	Runge-Kutta scheme order
n_{sub}	number of sub-increments
p'	isotropic effective pressure, M $L^{-1}T^{-2}$
p_c	isotropic preconsolidation pressure, M $L^{-1}T^{-2}$
POP	pre-overburden pressure, M $L^{-1}T^{-2}$
q	deviatoric stress, M $L^{-1}T^{-2}$
$\mathbf{r}$	force residual, M LT^{-2}
s	number of stages in Runge-Kutta scheme
t	time at the end of a loading increment
t_o	time at the start of a loading increment
t_{ref}	reference time
T	dimensionless time
v_o	initial specific volume

Greek

α	factor to control the size of load increment growth
α_{kj}	Runge-Kutta scheme coefficient
β	factor to control substep size
χ	safety factor to control the size of load increment
κ	unloading-reloading index
μ	Poisson's ratio
$\boldsymbol{\sigma}'$	effective Cauchy stress tensor, M $L^{-1}T^{-2}$
λ	plastic compression index
ω_k	Runge-Kutta scheme coefficient
$\Delta\mathbf{f}$	force increment, M LT^{-2}
Δt	time increment
ΔT	dimensionless time increment
ΔT_{min}	minimum dimensionless time increment
$\Delta\mathbf{u}$	displacement increment vector, L

alternative when the classical Newton-Raphson scheme fails, they still require the initial guess to be close to the correct solution to ensure convergence.

Gens and Potts (1988) carried out an early study on the different methods used to solve nonlinear equations in geotechnical applications with a constitutive model based on the critical state concept. The study included, in addition to the iterative procedures, the explicit first-order forward Euler scheme (tangent stiffness method). They concluded that there is no generic statement on the recommended method to be used and it is necessary to equip the numerical code with different solution methods so that the most suitable one can be used depending on the solved problem. In a later study, however, Potts and Ganendra (1992) concluded that the modified Newton-Raphson method is the most robust and economical method, which is now the standard in most of the Finite Element codes. To improve the convergence of the scheme, several numerical techniques can be used, including the arc length control and the line search method (Bathe, 2006; Crisfield, 1983).

Abbo and Sloan (1996) provided an improved second-order explicit scheme (Euler-modified Euler) with error control of the drift from the load path. Although, the scheme showed to be robust and stable even in applications that involve critical state soil models (Sheng and Sloan, 2001), the iterative methods were significantly faster, hence the scheme was not widely adopted. Recently, the numerical software started including the automatic (algorithmic) differentiation (AD) (Griewank and Walther, 2008), which means that the accurate derivation of the stiffness matrix is available without extra numerical burden. This opens new possibilities for novel numerical algorithms, more stable and robust than existing ones.

2. Importance and significance of the proposed method

This paper presents a novel alternative method which treats Eq. (1)

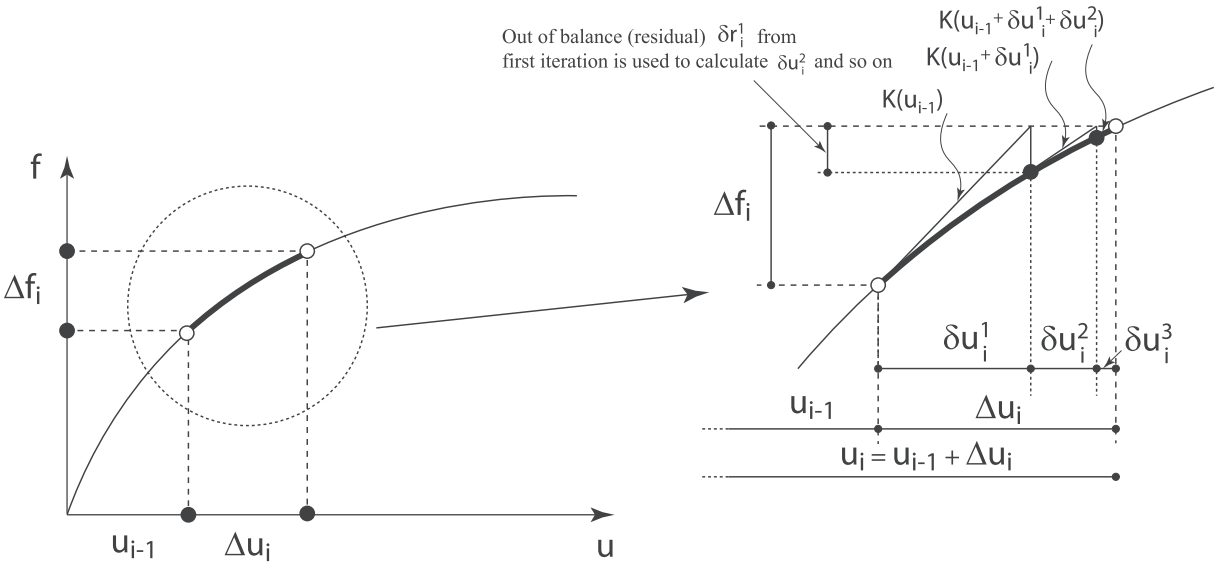

Fig. 1. Implicit methods: iterations per the i^{th} load increment to estimate the corresponding displacement increment.

as a differential equation and applies Runge-Kutta explicit schemes with automatic sub-incrementation (sub-stepping) and error control to solve it. The paper discusses first the theoretical background and the numerical implementation steps. Subsequently, the method is tested on a typical geotechnical problem of a shallow foundation resting on an elastoplastic soil. In the tests, the new method has been more stable than most common existing solutions. The proposed algorithm based on the 5th order Runge-Kutta scheme is also competitive with the iterative schemes regarding the time of calculations.

The new method of solution builds on a well-tested, robust scheme with mathematically proven convergence properties. Consequently, its adoption may be beneficial not only in the Finite Element software for geotechnical engineering, but in all Finite Element codes used for analysis of highly non-linear problems, for instance coupled thermo-hydro-mechanical-chemical analyses.

3. Solution algorithm

In the case of nonlinear material behaviour, the stiffness matrix in Equation (1) depends on stresses and strains. Those are related to displacements, hence the stiffness matrix is ultimately displacement dependent $\mathbf{K}(\mathbf{u}) = \partial \mathbf{f}_{\mathbf{int}}/\partial \mathbf{u}$, where $\mathbf{f}_{\text{int}}$ is the vector of internal nodal forces. This allows us to rewrite Equation (1) as a differential equation:

$$\mathrm{d}\mathbf{u}_i/\mathrm{d}t = [\mathbf{K}(\mathbf{u})]^{-1}\mathrm{d}\mathbf{f}_i/\mathrm{d}t \tag{2}$$

where $\mathrm{d}\mathbf{u}_i$ is the infinitesimal increment of displacements and $\mathrm{d}\mathbf{f}_i$ is the infinitesimal increment of externally applied forces over an algorithmic time infinitesimal increment $\mathrm{d}t$. The dependency of stiffness matrix on displacements stems from the complex constitutive behaviour of soil (material) which, in most cases, involves nonlinear elasticity and plasticity. Typically, Eq. (2) is solved incrementally allowing for linearization by applying the total external force in increments $\Delta\mathbf{f}_i$ per each time increment Δt. To enhance the mathematical clarity of Eq. (2), one introduces the dimensionless time $\mathrm{T} = (t - t_0)/\mathrm{d}t$ where t and t_0 are the algorithmic time at the end and at the start of the load increment resulting in $0 \leq \mathrm{T} \leq 1$. Then, following the chain rule, the incremental displacements per time increment will be $\mathrm{d}\mathbf{u}_i/\mathrm{d}t = \mathrm{d}\mathbf{u}_i/\mathrm{d}\mathrm{T} \times 1/\mathrm{d}t$. Substitution in Equation (2) yields:

$$\mathrm{d}\mathbf{u}_i/\mathrm{d}\mathrm{T} = [\mathbf{K}(\mathbf{u}_i)]^{-1}\mathrm{d}\mathbf{f}_i \tag{3}$$

Eq. (3) is a first-order ordinary nonlinear differential equation which can be solved using iterative methods (*implicit*) such as the Newton-Raphson method. Other possibility is to use the *explicit* methods that solve the equation directly without iterations but assume load increments $\Delta\mathbf{f}_i$ that satisfy certain accuracy condition. Fig. 1 and Fig. 2 show, for a one-dimensional case, how both methods approach the solution $\mathbf{u}_i$ for the i^{th} load increment $\Delta\mathbf{f}_i$. Based on Eq. (3), for each load increment $\Delta\mathbf{f}_i$, one calculates the corresponding displacement increment $\Delta\mathbf{u}_i$ and adds this value to the previously calculated total displacements $\mathbf{u}_{i-1}$ to determine the new $\mathbf{u}_i$. The solution starts with known initial displacement $\mathbf{u}_0$, thus Eq. (3) is an initial value problem.

Despite the rapid convergence rate of the implicit methods, their convergence is conditional as the initial prediction needs to be sufficiently close to the correct solution. Therefore, the Newton-Raphson method divergence happens especially often in the case of material behaviour with strong nonlinearity. Another major disadvantage of the implicit methods is that they do not have any load path error control meaning that it could converge if big loading increments are adopted, but to a wrong answer (Abbo and Sloan, 1996). On the other hand, the explicit methods are stable; however, the only explicit method commonly used in the past, the Forward Euler method, tends to drift away from the correct solution with the advancement of the solution. To control the drift, small load increments should be adopted which slow down the solution tremendously and make the Forward Euler method unattractive.

Although the explicit Runge-Kutta methods received wide recognition for integrating constitutive laws on the Gauss point level (Sloan, 1987; Sloan et al., 2001; Sołowski and Gallipoli, 2010), they got little attention on their applicability for solving the global finite element equations. Apart from the contribution by Abbo and Sloan (1996) who proposed an explicit scheme with error control that can be considered as a second-order scheme when compared to the first-order Forward Euler scheme, the authors are not aware of any other major contribution.

When solving Eq. (3) with Runge-Kutta method, the estimation of $\Delta\mathbf{u}_i$ is taken as a weighted sum of explicit evaluations of $\Delta\mathbf{u}_i^{(k)}$ at predefined positions k along the given algorithmic time interval $\Delta\mathrm{T}$. The number of these evaluations is directly related to the required accuracy of the solution, yielding a scheme with different orders (e.g. first, second … fifth and higher). Employing a high order Runge-Kutta scheme leads to a fast convergence and an accurate solution with well controlled errors. It should be noted that the Runge-Kutta method is a standard numerical method that can be reviewed elsewhere, see Lee and Schiesser (2003) for example.

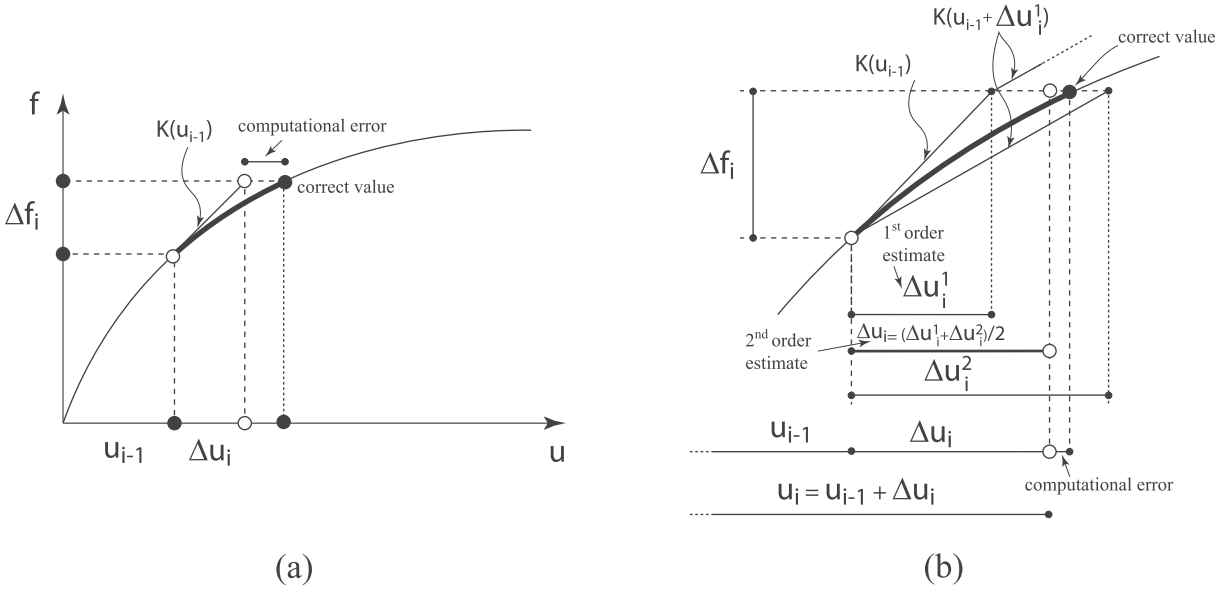

Fig. 2. Explicit methods: (a) first-order forward Euler and (b) second-order estimate of displacement increment per the i^{th} load increment.

Table 1
Stages for the 2nd order scheme.

Item	Estimate
Stage 1 displacement estimate:	$\Delta\mathbf{u}_i^{(1)} = [\mathbf{K}(\mathbf{u}_{i-1})]^{-1}\Delta\mathbf{f}_i$
Stage 2 displacement estimate:	$\Delta\mathbf{u}_i^{(2)} = [\mathbf{K}(\mathbf{u}_{i-1} + \Delta\mathbf{u}_i^{(1)})]^{-1}\Delta\mathbf{f}_i$
Second order displacement estimate:	$\mathbf{u}_i^{(2\mathrm{nd})} = \mathbf{u}_{i-1} + \frac{1}{2}(\Delta\mathbf{u}_i^{(1)} + \Delta\mathbf{u}_i^{(2)})$
Error	$\mathbf{Err}^{(2\mathrm{nd})} = \frac{1}{2}(\Delta\mathbf{u}_i^{(2)} - \Delta\mathbf{u}_i^{(1)})$

4. Runge-Kutta explicit scheme for load-deflection estimation

The first-order Runge-Kutta method is equivalent to the forward explicit Euler method where the displacement increment is the direct outcome of Eq. (3) with $d\mathrm{T} = 1$, see Fig. 2 and Eq. (4). Obviously, the accuracy of the first-order scheme is heavily dependent on the load increment size. To achieve a sufficiently accurate solution, very small increments should be adopted leading to a numerically expensive solution. For Runge-Kutta second-order scheme, the estimation of the displacement increment $\Delta\mathbf{u}_i$ needs two stages ($k = 1, 2$) of secondary displacement increment evaluations $\Delta\mathbf{u}_i^{(k)}$ (see Fig. 2 for a graphical clarification):

Stage $k = 1$:

$$\Delta\mathbf{u}_i^{(1)} = [\mathbf{K}(\mathbf{u}_{i-1}^{(1)})]^{-1}\Delta\mathbf{f}_i \tag{4}$$

Stage $k = 2$:

$$\Delta\mathbf{u}_i^{(2)} = [\mathbf{K}(\mathbf{u}_{i-1} + \Delta\mathbf{u}_i^{(1)})]^{-1}\Delta\mathbf{f}_i \tag{5}$$

where $\mathbf{K}$ is the displacement-dependent tangent stiffness matrix with $\mathbf{K}(\mathbf{u}_{i-1}^{(1)}) = \partial\mathbf{f}_{int}^{(1)}/\partial\mathbf{u}_{i-1}$ and $\mathbf{K}(\mathbf{u}_{i-1} + \Delta\mathbf{u}_i^{(1)}) = \partial\mathbf{f}_{int}^{(2)}/\partial(\mathbf{u}_{i-1} + \Delta\mathbf{u}_i^{(1)})$. Note that $\mathbf{f}_{int}^{(1)}$ and $\mathbf{f}_{int}^{(2)}$ represent the internal forces at the corresponding displacements $\mathbf{u}_{i-1}$ and $\mathbf{u}_{i-1} + \Delta\mathbf{u}_i^{(1)}$, respectively. Consequently, the second order estimation of the displacement increment $\Delta\mathbf{u}_i$ is:

$$\Delta\mathbf{u}_i = \frac{1}{2}\Delta\mathbf{u}_i^{(1)} + \frac{1}{2}\Delta\mathbf{u}_i^{(2)} = \frac{1}{2}(\Delta\mathbf{u}_i^{(1)} + \Delta\mathbf{u}_i^{(2)}) \tag{6}$$

As can be noticed, the weighing factors are identical here for the two estimations and equal ½. One important aspect to mention is that several Runge-Kutta methods have embedded a lower order solution, which can be used to estimate the error. This error is given as the difference between the current scheme displacement estimation and the estimation with the one order lower scheme. For example, for the second-order scheme presented above, the error $\mathbf{Err}$ is given as:

$$\mathbf{Err}^{(2\mathrm{nd})} = \mathbf{u}^{(2\mathrm{nd})} - \mathbf{u}^{(1\mathrm{st})} = \underbrace{\left[\mathbf{u}_{i-1} + \frac{1}{2}(\Delta\mathbf{u}_i^{(1)} + \Delta\mathbf{u}_i^{(2)})\right]}_{\text{second order estimate}} - \underbrace{[\mathbf{u}_{i-1} + \Delta\mathbf{u}_i^{(1)}]}_{\text{first order estimate}}$$
$$= \frac{1}{2}(\Delta\mathbf{u}_i^{(2)} - \Delta\mathbf{u}_i^{(1)}) \tag{7}$$

Runge-Kutta schemes with higher order of accuracy follow a similar algorithm. By knowing the correct weights and positions to estimate each secondary stage evaluation, the displacement increment is determined as a weighted sum of the performed evaluations. For an arbitrary high order method, Eq. (4) and Eq. (5) read:

$$\Delta\mathbf{u}_i^{(k)} = [\mathbf{K}(\mathbf{u}_i^{(k)})]^{-1}\Delta\mathbf{f}_i \tag{8}$$

where

$$\mathbf{u}_i^{(k)} = \mathbf{u}_{i-1} + \sum_{j=1}^{k-1} \alpha_{kj}\Delta\mathbf{u}_i^{(j)} \tag{9}$$

Here $\mathbf{u}_i^{(k)}$ represents the updated displacements according to the adopted Runge-Kutta scheme. The stiffness matrix $\mathbf{K}(\mathbf{u}_i^{(k)}) = \partial\mathbf{f}_{int}^{(k)}(\mathbf{u}_i^{(k)})/\partial\mathbf{u}_i^{(k)}$ where $\mathbf{f}_{int}^{(k)}(\mathbf{u}_i^{(k)})$ indicates the corresponding displacement dependent internal forces to the displacement's evaluation $\mathbf{u}_i^{(k)}$ at the calculation stage k. In this contribution, the evaluation

Table 2
Stages for the 3rd order scheme.

Item	Estimate
Stage 1 displacement estimate:	$\Delta\mathbf{u}_i^{(1)} = [\mathbf{K}(\mathbf{u}_{i-1})]^{-1}\Delta\mathbf{f}_i$
Stage 2 displacement estimate:	$\Delta\mathbf{u}_i^{(2)} = \left[\mathbf{K}\left(\mathbf{u}_{i-1} + \frac{2}{3}\Delta\mathbf{u}_i^{(1)}\right)\right]^{-1}\Delta\mathbf{f}_i$
Stage 3 displacement estimate:	$\Delta\mathbf{u}_i^{(3)} = \left[\mathbf{K}\left(\mathbf{u}_{i-1} + \frac{2}{3}\Delta\mathbf{u}_i^{(2)}\right)\right]^{-1}\Delta\mathbf{f}_i$
Third order displacement estimate:	$\mathbf{u}_i^{(3\mathrm{rd})} = \mathbf{u}_{i-1} + \frac{2}{8}\Delta\mathbf{u}_i^{(1)} + \frac{3}{8}\Delta\mathbf{u}_i^{(2)} + \frac{3}{8}\Delta\mathbf{u}_i^{(3)}$
Error	$\mathbf{Err}^{(3\mathrm{rd})} = \frac{3}{8}(\Delta\mathbf{u}_i^{(3)} - \Delta\mathbf{u}_i^{(2)})$

of the stiffness matrix $\mathbf{K}(\mathbf{u}_i^{(k)})$ is carried out using the automatic differentiation (AD) during the finite elements assembly process as will be discussed later. The estimated displacements are then calculated as:

$$\mathbf{u}_i^{(n)} = \mathbf{u}_{i-1} + \sum_{k=1}^{s} \omega_k\Delta\mathbf{u}_i^{(k)} \tag{10}$$

where s is the total number of evaluation stages that is dependent on the scheme order n (Lee and Schiesser, 2003).

By subtracting the lower order solution from the one order higher solution, the error can be estimated as:

$$\mathbf{Err}^{(n)} = \sum_{k=1}^{s} \lambda_k\Delta\mathbf{u}_i^{(k)} \tag{11}$$

Tables 1–6 give the coefficients α_{kj}, ω_k and λ_k for Eqs. (9), (10) and (11) for Runge-Kutta schemes of 2nd, 3rd, 4th and 5th order. Note that any Runge-Kutta scheme, e.g. those given in (Lee and Schiesser, 2003), can be used – tables give only the coefficients for schemes tested in this paper.

5. Runge-Kutta explicit scheme with error control

The error in the Runge-Kutta scheme is dependent on the size of the load increment $\Delta\mathbf{f}_i$. For the same increment size, the higher-order schemes yield predictions that are more accurate. Additionally, the method assesses the error which can be used to automatically choose the next load increment size, so that the error is bound within a desirable range, similarly as in the case of stress integration (Abbo and Sloan, 1996; Sloan, 1987; Sołowski and Gallipoli, 2010). The proposed steps of the method to solve the global finite element equations are illustrated in Diagram 1 and listed in Table 7.

The scheme starts by assuming that the first load sub-increment $\Delta\mathbf{f}_j = \Delta\mathrm{T}_\mathrm{j} \times \Delta\mathbf{f}_\mathrm{i}$, where $\mathrm{j} = 1$ is the number of the current sub-increment, equals the full load increment $\Delta\mathbf{f}_\mathrm{i}$ by using an initial sub-increment size of $\Delta\mathrm{T}_\mathrm{j} = 1$. After calculating the corresponding displacement using $\Delta\mathbf{f}_j$ by employing Eq. (9) and Eq. (10), the occurred relative error *Error* is estimated by applying Eq. (11). If this error is less or equal to the imposed tolerance D_{Tol} by the user, the current sub-increment is accepted and the program moves to the next load increment (see Diagram 1 and Table 7). On the other hand, if the error is greater than D_{Tol}, the current sub-increment is rejected and instead of $\Delta\mathrm{T}_\mathrm{j} = 1$, a new smaller value is estimated using Eq. (12). At this point the calculations are repeated with the new sub-increment size. The algorithm will continue reducing the increment size until the error criterion is satisfied and the step is accepted.

For both accepted and rejected sub-increments, a new sub-increment size is estimated again using Eq. (12). The procedure continues until the full increment is applied yielding $\Delta\mathbf{f}_i = \sum_{j=1}^{n_{sub}} \Delta\mathbf{f}_\mathrm{j}$ where $\Delta\mathbf{f}_j = \Delta\mathrm{T}_\mathrm{j} \times \Delta\mathbf{f}_\mathrm{i}$, $\sum_{j=1}^{n_{sub}} \Delta\mathrm{T}_\mathrm{j} = 1$ and n_{sub} is the resulted total number of accepted sub-increments.

The size of the new sub-increment $\Delta\mathrm{T}_{new}$ is estimated based on the old given sub-increment size $\Delta\mathrm{T}_{old}$, the scheme order of accuracy n and the estimated error *Error* in the current sub-increment. Thus, the new

Table 3
Stages for the 4th order scheme.

Item	Estimate
Stage 1 displacement estimate:	$\Delta\mathbf{u}_i^{(1)} = [\mathbf{K}(\mathbf{u}_{i-1})]^{-1}\Delta\mathbf{f}_i$
Stage 2 displacement estimate:	$\Delta\mathbf{u}_i^{(2)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{21}\Delta\mathbf{u}_i^{(1)})]^{-1}\Delta\mathbf{f}_i$
Stage 3 displacement estimate:	$\Delta\mathbf{u}_i^{(3)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{31}\Delta\mathbf{u}_i^{(1)} + \alpha_{32}\Delta\mathbf{u}_i^{(2)})]^{-1}\Delta\mathbf{f}_i$
Stage 4 displacement estimate:	$\Delta\mathbf{u}_i^{(4)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{41}\Delta\mathbf{u}_i^{(1)} + \alpha_{42}\Delta\mathbf{u}_i^{(2)} + \alpha_{43}\Delta\mathbf{u}_i^{(3)})]^{-1}\Delta\mathbf{f}_i$
Fourth order displacement estimate:	$\mathbf{u}_i^{(4\mathbf{th})} = \mathbf{u}_{i-1} + \omega_1\Delta\mathbf{u}_i^{(1)} + \omega_2\Delta\mathbf{u}_i^{(2)} + \omega_3\Delta\mathbf{u}_i^{(3)} + \omega_4\Delta\mathbf{u}_i^{(4)}$
Error	$\mathbf{Err}^{(4\mathbf{th})} = \lambda_1\Delta\mathbf{u}_i^{(1)} + \lambda_2\Delta\mathbf{u}_i^{(2)} + \lambda_3\Delta\mathbf{u}_i^{(3)} + \lambda_4\Delta\mathbf{u}_i^{(4)}$

*Table 4 lists the corresponding coefficients.

Table 4
Coefficients for the 4th order scheme.

Item	Coefficient value			
Stage 2	α_{21}			
	−0.4			
Stage 3	α_{31}	α_{32}		
	0.6684895833	−0.2434895833		
Stage 4	α_{41}	α_{42}	α_{43}	
	−2.323685857	1.125483559	2.198202298	
Displacement	ω_1	ω_2	ω_3	ω_4
	0.03431372549	0.02705627706	0.7440130202	0.1946169772
Error	λ_1	λ_2	λ_3	λ_4
	0.03431372549	−0.01262626262	−0.0289338397	0.00724637679

Table 5
Stages for the 5th order scheme.

Item	Estimate
Stage 1 displacement estimate:	$\Delta\mathbf{u}_i^{(1)} = [\mathbf{K}(\mathbf{u}_{i-1})]^{-1}\Delta\mathbf{f}_i$
Stage 2 displacement estimate:	$\Delta\mathbf{u}_i^{(2)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{21}\Delta\mathbf{u}_i^{(1)})]^{-1}\Delta\mathbf{f}_i$
Stage 3 displacement estimate:	$\Delta\mathbf{u}_i^{(3)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{31}\Delta\mathbf{u}_i^{(1)} + \alpha_{32}\Delta\mathbf{u}_i^{(2)})]^{-1}\Delta\mathbf{f}_i$
Stage 4 displacement estimate:	$\Delta\mathbf{u}_i^{(4)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{41}\Delta\mathbf{u}_i^{(1)} + \alpha_{42}\Delta\mathbf{u}_i^{(2)} + \alpha_{43}\Delta\mathbf{u}_i^{(3)})]^{-1}\Delta\mathbf{f}_i$
Stage 5 displacement estimate:	$\Delta\mathbf{u}_i^{(5)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{51}\Delta\mathbf{u}_i^{(1)} + \alpha_{52}\Delta\mathbf{u}_i^{(2)} + \alpha_{53}\Delta\mathbf{u}_i^{(3)} + \alpha_{54}\Delta\mathbf{u}_i^{(4)})]^{-1}\Delta\mathbf{f}_i$
Stage 6 displacement estimate:	$\Delta\mathbf{u}_i^{(6)} = [\mathbf{K}(\mathbf{u}_{i-1} + \alpha_{61}\Delta\mathbf{u}_i^{(1)} + \alpha_{62}\Delta\mathbf{u}_i^{(2)} + \alpha_{63}\Delta\mathbf{u}_i^{(3)} + \alpha_{64}\Delta\mathbf{u}_i^{(4)} + \alpha_{65}\Delta\mathbf{u}_i^{(5)})]^{-1}\Delta\mathbf{f}_i$
Fifth order displacement estimate:	$\mathbf{u}_i^{(5\mathbf{th})} = \mathbf{u}_{i-1} + \omega_1\Delta\mathbf{u}_i^{(1)} + \omega_2\Delta\mathbf{u}_i^{(2)} + \omega_3\Delta\mathbf{u}_i^{(3)} + \omega_4\Delta\mathbf{u}_i^{(4)} + \omega_5\Delta\mathbf{u}_i^{(5)} + \omega_6\Delta\mathbf{u}_i^{(6)}$
Error	$\mathbf{Err}^{(5\mathbf{th})} = \lambda_1\Delta\mathbf{u}_i^{(1)} + \lambda_2\Delta\mathbf{u}_i^{(2)} + \lambda_3\Delta\mathbf{u}_i^{(3)} + \lambda_4\Delta\mathbf{u}_i^{(4)} + \lambda_5\Delta\mathbf{u}_i^{(5)} + \lambda_6\Delta\mathbf{u}_i^{(6)}$

* Table 6 lists the corresponding coefficients in the case of 5th order scheme.

Table 6
Coefficients for the 5th order scheme.

Item	Coefficient value					
Stage 2	α_{21}					
	1/5					
Stage 3	α_{31}	α_{32}				
	3/40	9/40				
Stage 4	α_{41}	α_{42}	α_{43}			
	3/10	−9/10	6/5			
Stage 5	α_{51}	α_{52}	α_{53}	α_{54}		
	−11/54	5/2	−70/27	35/27		
Stage 6	α_{61}	α_{62}	α_{63}	α_{64}	α_{65}	
	1631/55296	175/512	575/13824	44275/110592	253/4096	
Displacement	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6
	37/378	0	250/621	125/594	0	512/1771
Error	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
	−277/64512	0	6925/370944	−6925/202752	−277/14336	277/7084

sub-increment size is $\Delta T_{new} = \beta \times \Delta T_{old}$ where:

$$\beta = \chi\sqrt[n]{\frac{D_{Tol}}{Error}} \tag{12}$$

Here, β is the relative change in the next sub-increment size, D_{Tol} is the required accuracy imposed by the user and χ is a safety factor, as this sub-increment size estimation is approximate. Such a choice of the new sub-increment size should keep the error in the next step below the tolerated value (Gustafsson, 1992). To reduce the chance of spurious increase/decrease of the steps due to random correlation between the

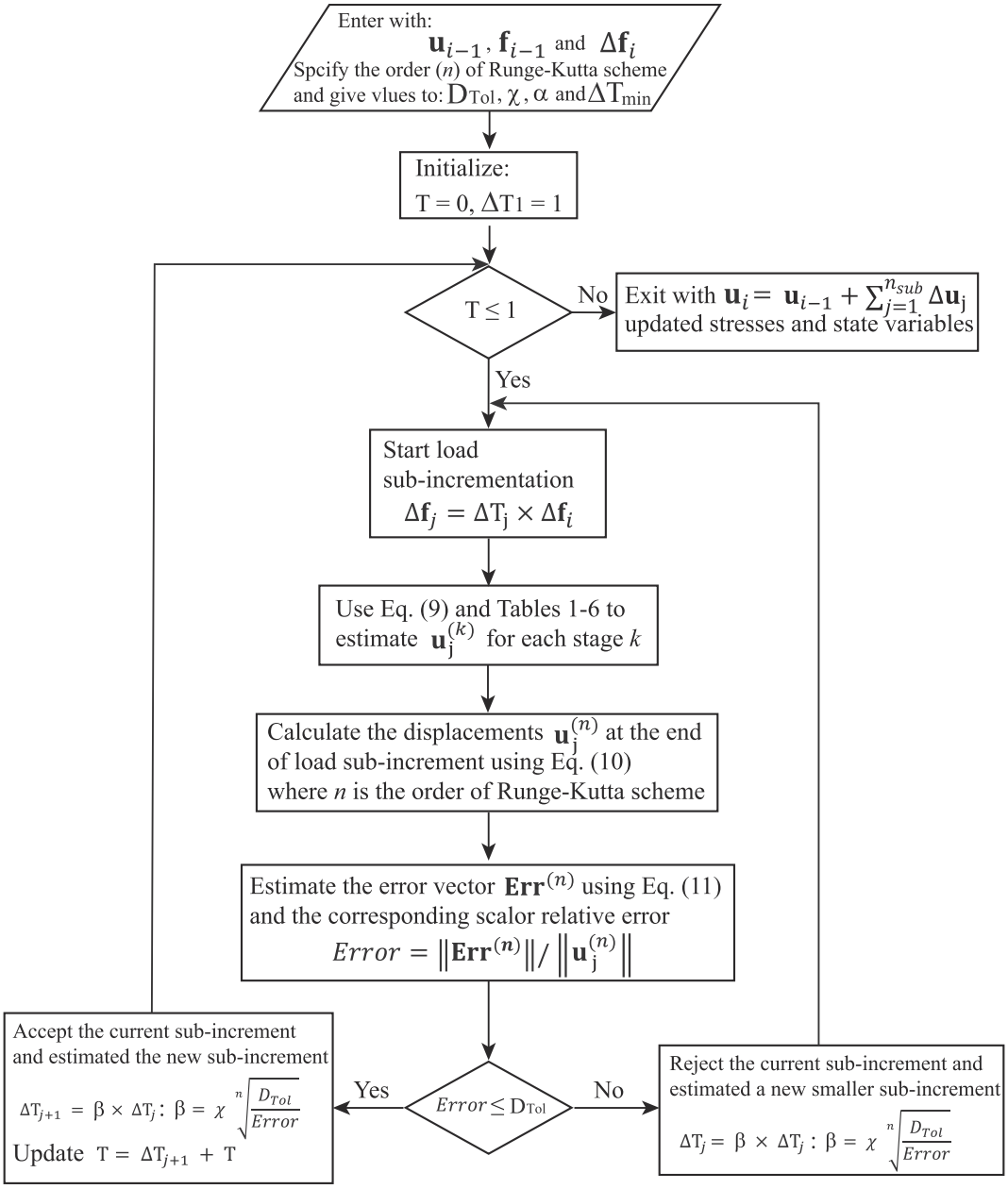

Diagram 1. A flow chart illustrates the required steps by the proposed Runge-Kutta scheme.

two solutions used for estimation of the error and to avoid unnecessary failed sub-increment sizes, the resulted β from Eq. (12) is limited by value of α and 0.1. Previous experience (Abbo and Sloan, 1996; Sloan et al., 2001) showed that the recommended values are in the range $0.7 \leq \chi \leq 0.9$ and $1.1 \leq \alpha \leq 2.0$. The authors used the values $\chi = 0.9$ and $\alpha = 1.2$, leading to $0.1 \leq \beta \leq 1.2$ in the calculations. This means that the increase of the next load sub-increment is capped at 120% of the current load sub-increment. As can be interpreted from Eq. (12), the new sub-increment size will be reduced in case of rejected step ($Error > D_{Tol}$).

6. Note on the evaluation of the global stiffness matrix by automatic differentiation

The evaluation of the global stiffness matrix is an important step during the solution for displacement increment as given by Eq. (8). Typically, that would need the assembly of the global stiffness matrix from the individual finite element stiffness matrices respecting the relevant degrees of freedom (Smith et al., 2013). Thebes code (Abed and Sołowski, 2017, 2019), based on Numerrin numerical solver (Laitinen, 2013), follows a different method to evaluate the stiffness matrix by the direct differentiation of the internal nodal forces with respect to the current nodal displacements. That is carried out using the automatic (algorithmic) differentiation (AD) (Griewank and Walther, 2008) where the stiffness matrix is evaluated during the assembly of the internal nodal forces. The main idea behind AD is that any function is executed as a sequence of basic operations (e.g. additions, subtractions, multiplications, etc.). By applying the chain rule repeatedly to these operations, the derivative of such a function can be computed with no truncation errors (i.e. to the machine precision). In order to achieve that, a parallel program translates the code of the respective function (that is not necessarily present in closed form but most of the time as a computer program) into a new code that contains derivatives. The application of this method is not well recognized in the field of computational geomechanics but has received full consideration in the general field of the finite element method (Tijskens et al., 2002; Rothe and

Table 7
Required steps to integrate the global finite element equations using the Runge-Kutta scheme.

1. Enter with the estimated total displacements from the previous loading increment $\mathbf{u}_{i-1}$, the applied load $\mathbf{f}_{i-1}$, the new load increment $\Delta\mathbf{f}_i$. Provide values for the tolerated error D_{Tol}, the minimum load increment size ΔT_{min}, the safety factor for load increment size χ and the threshold value to control the maximum growth of the next load increment α. Choose the required order of the Runge-Kutta scheme (n). Initialize $T = 0$ and $\Delta T_1 = 1$.
2. Loop as long as $T < = 1$ and perform *a* to *d*, otherwise, go to Step 3
   a. Based on the chosen order of Runge-Kutta scheme (n), evaluate the displacements for each stage k using formula (9) together with the corresponding coefficients in Tables 1–6. Use $\Delta\mathbf{f}_j = \Delta T_j \times \Delta\mathbf{f}_i$ in the formula. The subscript j indicates the number of the current sub-increment. Note that the global stiffness matrix at each stage k is evaluated directly by the algorithmic differentiation of the generated internal nodal forces $\mathbf{f}_{int}^{(k)}$ with respect to the corresponding updated displacements $\mathbf{u}_i^{(k)}$.
   b. Estimate the displacements at the end of the load sub-increment $\mathbf{u}_j^{(n)}$ using Eq. (10) depending on the scheme order (n).
   c. Estimate the error vector $\mathbf{Err}^{(n)}$ based on Eq. (11) depending on the chosen scheme order, then estimate the relative error:

$$Error = \|\mathbf{Err}^{(n)}\| / \|\mathbf{u}_j^{(n)}\|$$

   For the 2D case, the error norm $\|\mathbf{Err}^{(n)}\|$ is calculated as:

$$\|\mathbf{Err}^{(n)}\| = \sqrt{Err(0)_1^{(n)^2} + Err(1)_1^{(n)^2} + \cdots + Err(0)_m^{(n)^2} + Err(1)_m^{(n)^2}}$$

   where Err is the displacement error component being estimated, according to each order, based on Tables 1–6 and m is the total number of nodes in the mesh. Note that $Err(0)$ and $Err(1)$ represent the error in estimating the displacement increment of the calculated node in x and y direction, respectively.
   d. Check the condition $Error \leq D_{Tol}$:
      i. If yes, estimate the new load increment size ΔT_{j+1} based on the factor β, update $T = \Delta T_{j+1} + T$ and repeat from Step 2, where:

$$\beta = \chi \sqrt[n]{\frac{D_{Tol}}{Error}}; \quad \beta = \min(\beta, \alpha); \quad \Delta T_{j+1} = \beta \times \Delta T_j$$

      Apply the following constraints on the new sub-increment to avoid very small sizes or overloading once approaching the completion of the current load increment i.

$$\Delta T_{j+1} = \max(\Delta T_{j+1}, \Delta T_{min}); \quad \Delta T_{j+1} = \min(\Delta T_{j+1}, 1.0 - T)$$

      ii. If $Error > D_{Tol}$ then reject the current load sub-increment, estimate a new "smaller" load sub-increment $\Delta T_j = \beta \times \Delta T_j$ and repeat from Step 2.

$$\beta = \chi \sqrt[n]{\frac{D_{Tol}}{Error}}$$

      Apply the following constraints on the new reduced sub-increment to avoid very small sizes.

$$\beta = \max(\beta, 0.1); \quad \Delta T_j = \max(\beta \times \Delta T_j, \Delta T_{min})$$

3. Exit with the new displacements $\mathbf{u}_i^{(n)} = \mathbf{u}_{i-1} + \sum_{j=1}^{n_{sub}} \Delta\mathbf{u}_j$, update stresses and state variables and continue to the new load increment $i + 1$.

Table 8
Modified Cam Clay parameters of the modelled clay layer.

μ	κ	λ	M	v_o	POP [kPa]
0.3	0.02	0.2	1.0	2.3	20.0

Hartmann, 2015; Zwicke et al., 2016; Mitusch, 2018). The theoretical details of algorithmic differentiation are out of the scope of this paper, however the interested reader is referred to (Griewank and Walther, 2008; Bischof et al., 1992, 1996) for more details. In any case, the way the global stiffness matrix is determined does not affect the essence of the proposed method for solving the finite element equations.

7. Comparison of a single Newton-Raphson iteration versus one Runge-Kutta calculation stage

To assess the efficiency of the proposed scheme, we compare it to the Newton-Raphson scheme, typically used in Finite Element Method calculations. During a standard *full Newton-Raphson* iteration k within the load increment (step) i, the following operations are performed:

1. Form the tangent stiffness matrix $\mathbf{K}_T$ (Jacobian matrix) at the estimated displacement from previous iteration:

$$\mathbf{K}_T(\mathbf{u}_i^k) = \frac{\partial \mathbf{r}_i^k}{\partial \mathbf{u}_i^k}$$

2. Compute the improvement in displacement value $\delta\mathbf{u}_i^k$

$$\delta\mathbf{u}_i^k = [\mathbf{K}_T(\mathbf{u}_i^k)]^{-1}\mathbf{r}_i^k$$

3. Compute the new displacements $\mathbf{u}_i^{k+1}$ to be used in the next iteration $k + 1$

$$\mathbf{u}_i^{k+1} = \mathbf{u}_i^k - \delta\mathbf{u}_i^k$$

where $\mathbf{r} = \mathbf{f}_{ext} - \mathbf{f}_{int}$ is the force residual (out of balance) vector. The iterations continue until the imposed convergence criterion is satisfied.

On the other hand, the operations for one stage k of *Runge-Kutta* schemes follow the steps:

1. Evaluate the stiffness matrix for a stage k using the estimated displacements for that stage:

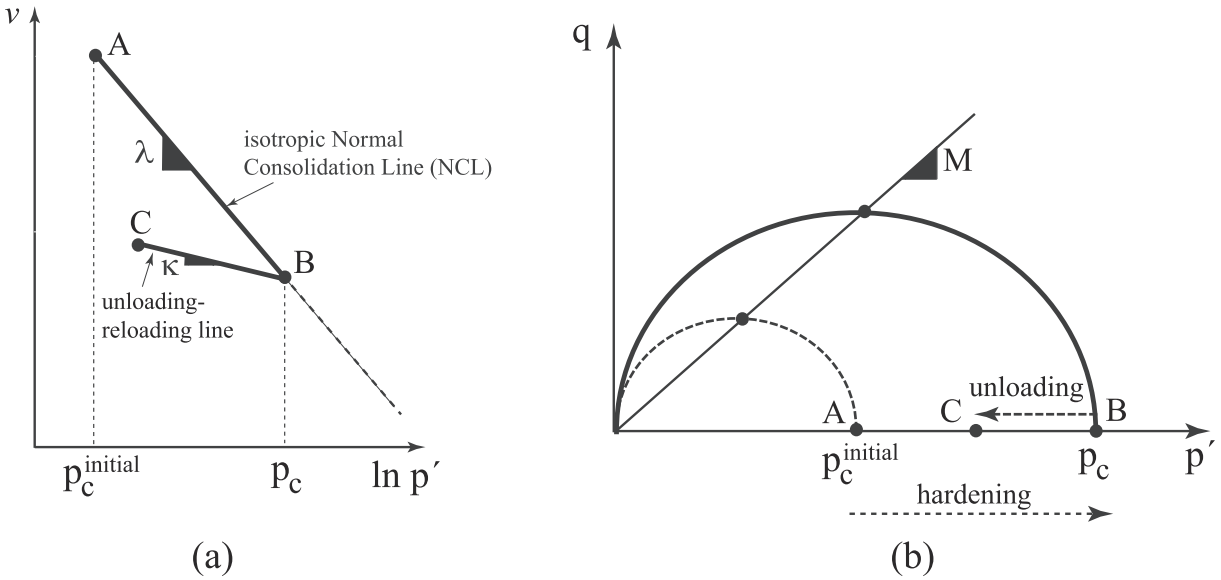

Fig. 3. Modified Cam Clay model: a) isotropic loading-unloading in v-ln p' plane; b) yield surface in p'-q plane.

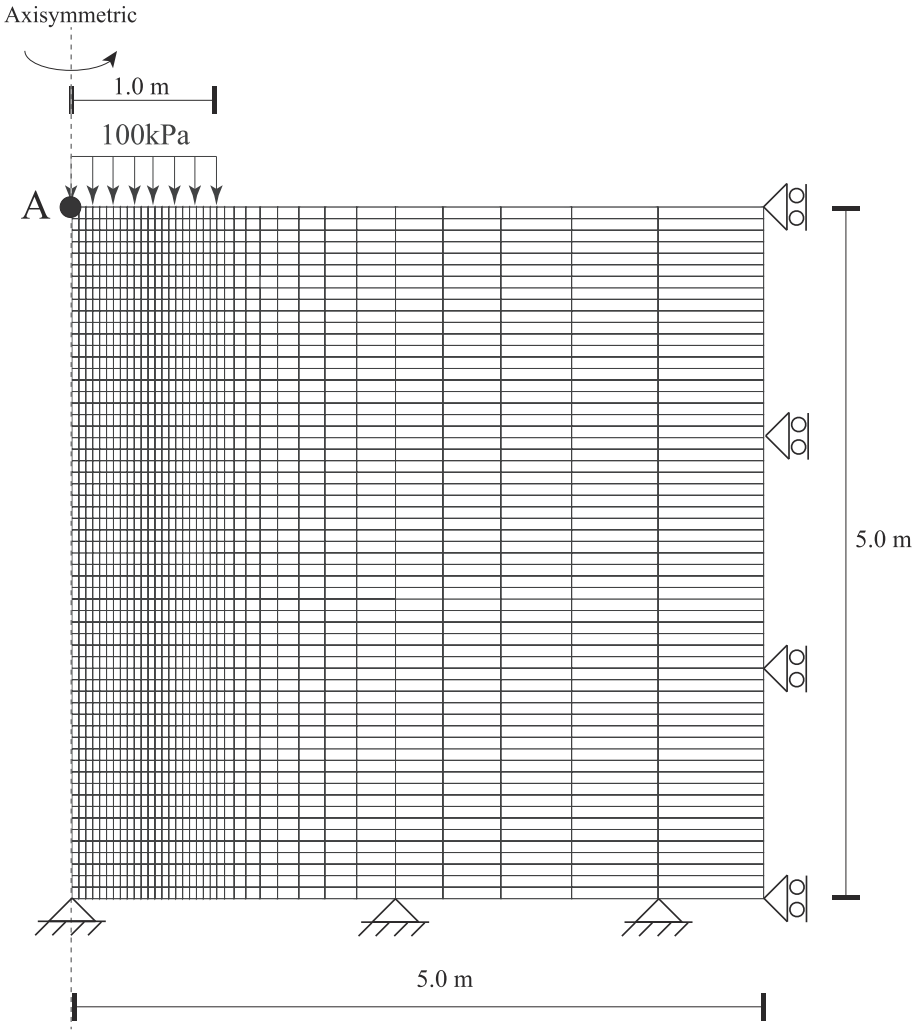

Fig. 4. Finite element model for the shallow footing example.

Table 9
Time and number of sub-increments required by each scheme for the shallow footing problem with $D_{Tol} = 1.0\text{E}-3$.

Scheme	No. successful sub-increments	No. failed sub-increments	Number of required stages/iterations	Calculation Time
5th order	17	1	108	$t_{ref}^{(3)}$
4th order	92	8	400	$3.2t_{ref}^{(3)}$
3rd order	821	10	2493	$22t_{ref}^{(3)}$
2nd order	–	–	–	–
Newton-Raphson	21	1	131	$\approx 1.25t_{ref}^{(3)}$ with divergence at 75 kPa

Table 10
Time and number of sub-increments required by each scheme for the shallow footing problem with $D_{Tol} = 1.0\text{E}-2$.

Scheme	No. successful sub-increments	No. failed sub-increments	Number of required stages/iterations	Calculation Time
5th order	9	0	54	$t_{ref}^{(2)}$
4th order	18	0	72	$1.15t_{ref}^{(2)}$
3rd order	70	3	219	$4.2t_{ref}^{(2)}$
2nd order	357	10	734	$15t_{ref}^{(2)}$

$$\mathbf{K}(\mathbf{u}_i^{(k)}) = \frac{\partial \mathbf{f}_{int}^{(k)}(\mathbf{u}_i^{(k)})}{\partial \mathbf{u}_i^{(k)}}$$

2. Calculate the displacement increment using the estimated stiffness matrix in Step 1 and the provided loading increment:

$$\Delta \mathbf{u}_i^{(k)} = [\mathbf{K}(\mathbf{u}_i^{(k)})]^{-1} \Delta \mathbf{f}_i$$

Table 11
Time and number of sub-increments required by each scheme for the shallow footing problem with $D_{Tol} = 1.0\text{E}-1$.

Scheme	No. successful sub-increments	No. failed sub-increments	Number of required stages/iterations	Calculation Time
5th order	9	0	54	$t_{ref}^{(1)}$
4th order	16	0	64	$1.03t_{ref}^{(1)}$
3rd order	17	0	51	$0.9t_{ref}^{(1)}$
2nd order	43	3	92	$1.9t_{ref}^{(1)}$

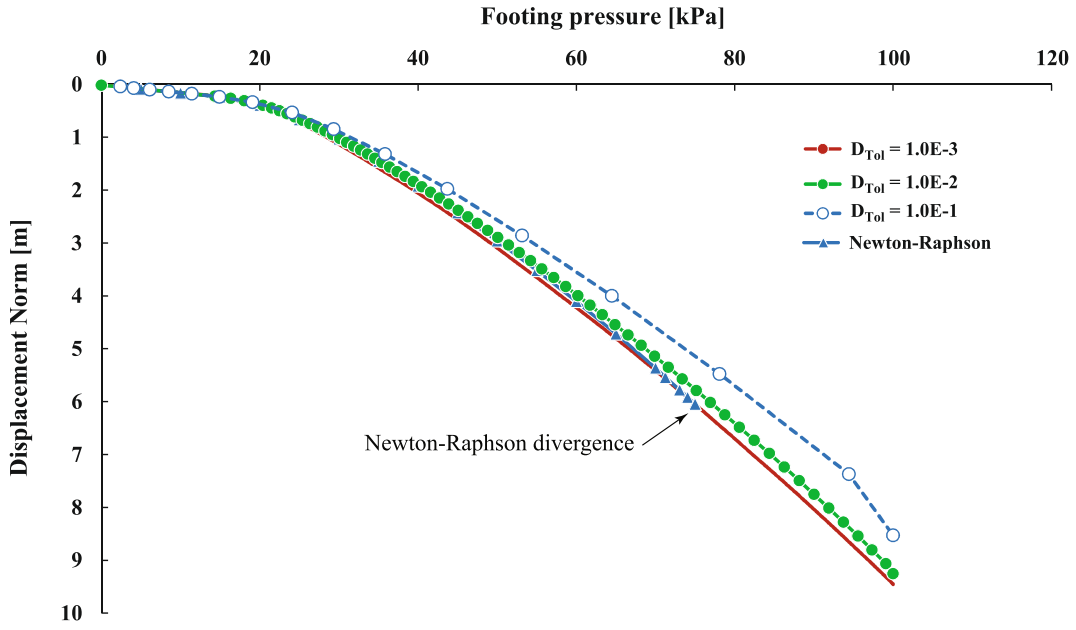

Fig. 5. Numerical results with 3rd order scheme and different tolerated errors.

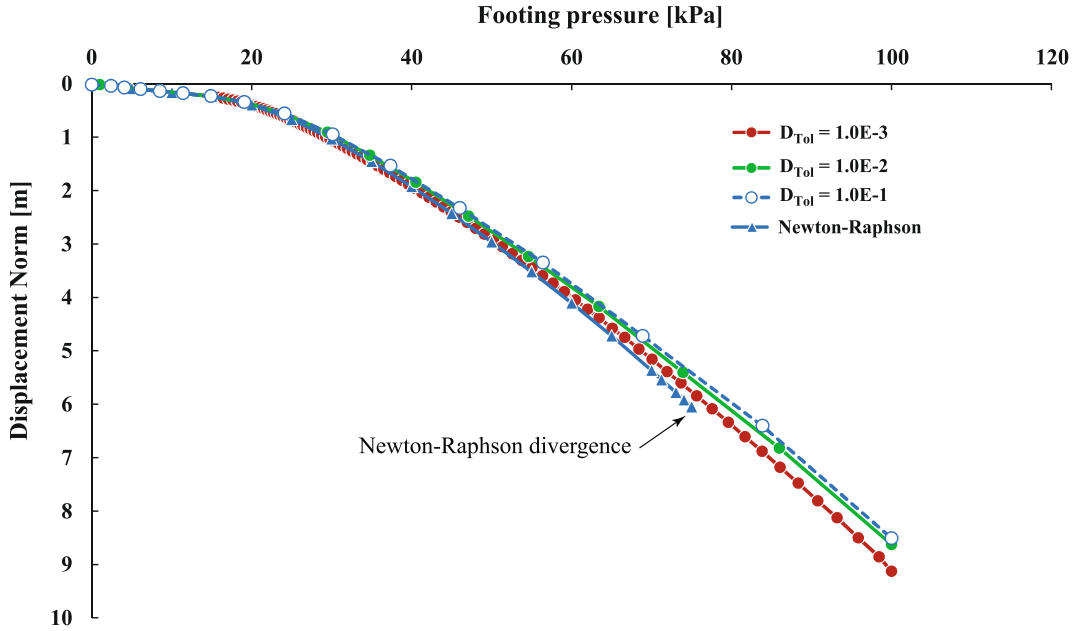

Fig. 6. Numerical results with 4th order scheme and different tolerated errors.

3. Calculate the displacements to be used in the next stage $k + 1$:

$$\mathbf{u}_i^{(k+1)} = \mathbf{u}_{i-1} + \sum_{j=1}^{k} \alpha_{kj} \Delta \mathbf{u}_i^{(j)}$$

By comparison, one notices that the number of computational operations in a single iteration of Newton-Raphson method is roughly comparable to that performed during one stage of a Runge-Kutta scheme (though the Runge-Kutta method does require more summation operations in Step 3 to estimate the displacements at the required

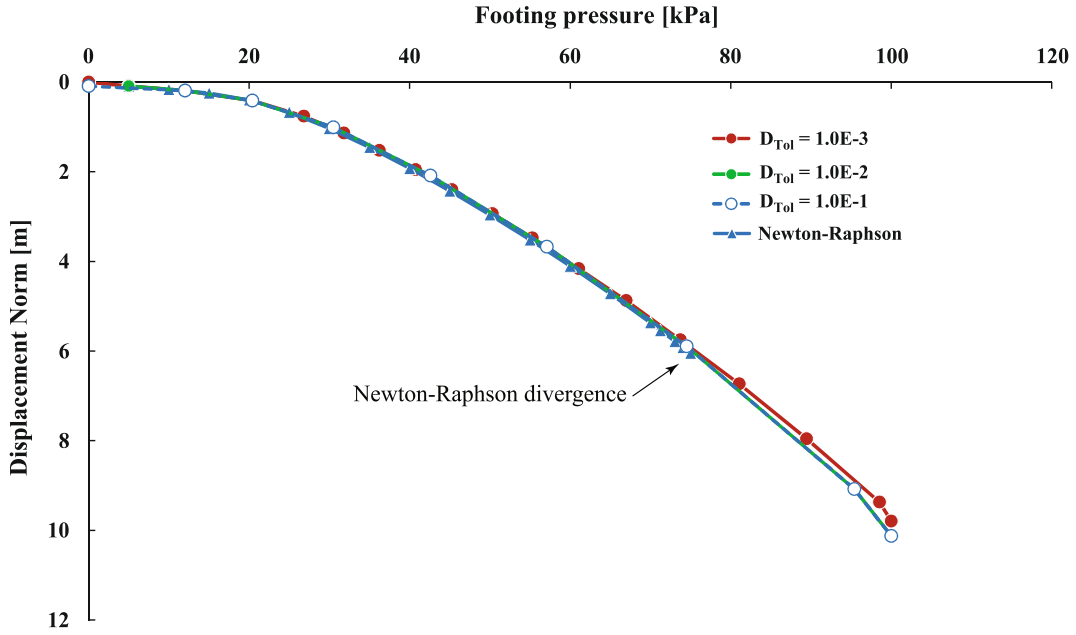

Fig. 7. Numerical results with 5th order scheme and different tolerated errors.

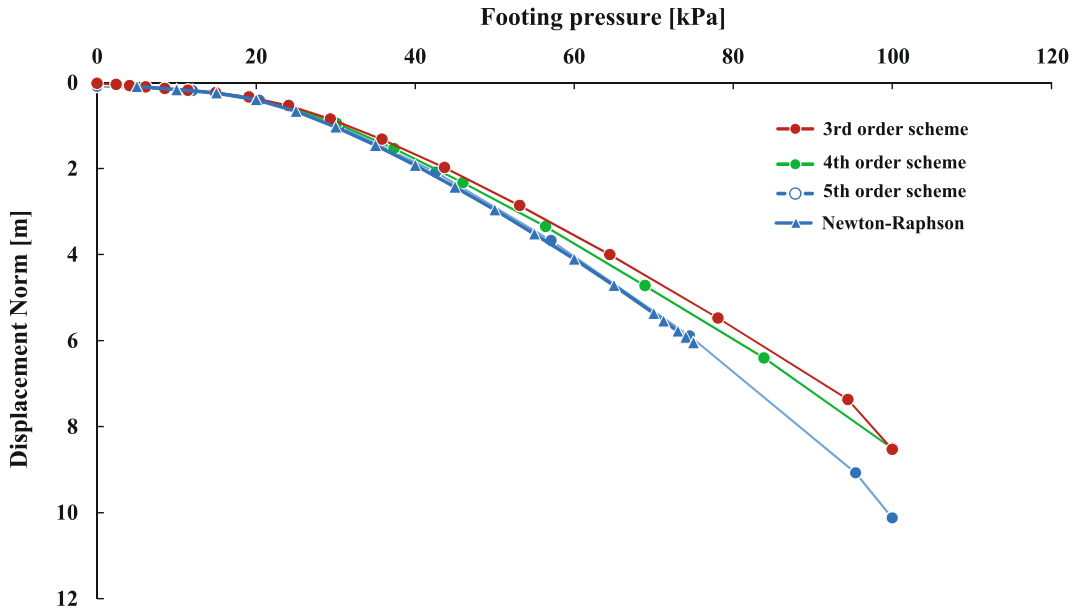

Fig. 8. Numerical results with different schemes and tolerated error (D_{Tol} = 1.0E−1).

position for evaluating the stiffness matrix). That gives us a way to approximately compare the computational time in both methods by comparing the number of required iterations and the required calculation stages in order to solve a given problem.

8. Numerical applications

The following examples discuss the performance of the proposed method based on simulations of the behaviour of a shallow foundation on an elastoplastic soil. The section provides simulations results and computation time with both the proposed schemes and the Newton-Raphson iterative scheme.

It is worth noting that the steps of the proposed method are independent of the adopted material behaviour model on Gauss point level. Thus, in principle, any suitable constitutive model for soil behaviour can be used including linear elasticity, nonlinear elasticity, elastoplastic etc. However, in this paper, in order to test the convergence of the method in case of a challenging material behaviour, the soil is modelled as Modified Cam Clay (MCC) which is an elastoplastic model based on the principles of critical state soil mechanics (Schofield and Wroth, 1968; Wood, 1990). Assuming a constant Poisson's ratio μ, the model requires the following soil parameters: 1) the initial specific volume v_0 at the reference isotropic effective pressure p' = 1 kPa; 2) the elastic unloading-reloading index κ to reproduce the elastic behaviour; 3) the plastic compression index λ to capture the behaviour during plasticity (both κ and λ are estimated in the plane v-ln p', see Fig. 3a); and 4) the slope of the critical state line M to predict failure (see Fig. 3b). The MCC is a volumetric hardening model where the hardening parameter is the isotropic preconsolidation pressure p_c and the yield function F is given as an ellipse in p'-q plane (see Fig. 3b):

$$F = q^2 - M^2 p'(p_c - p') = 0 \tag{13}$$

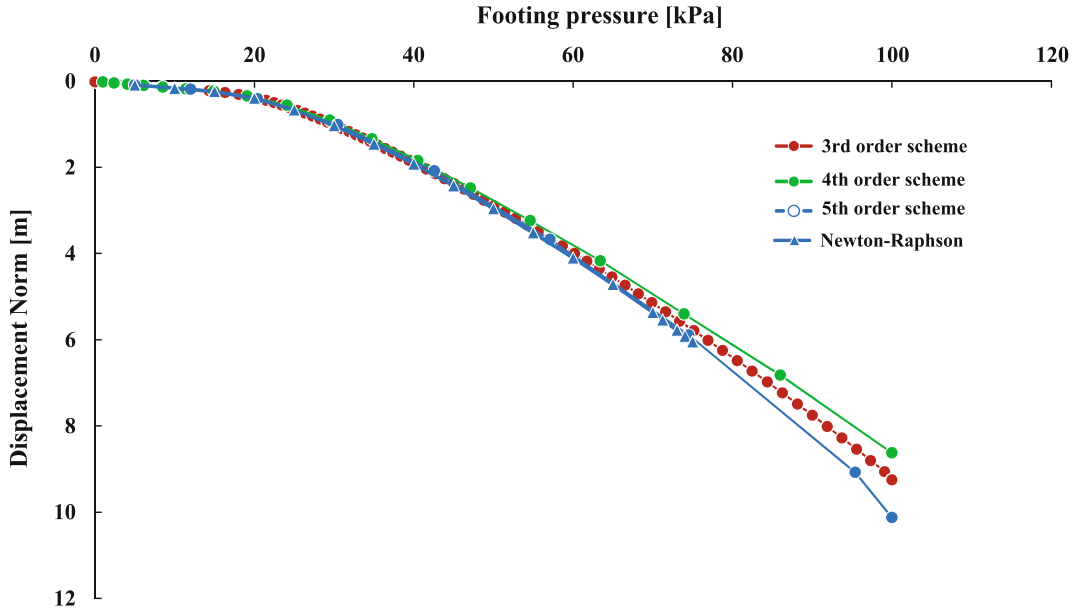

Fig. 9. Numerical results with different schemes and tolerated error (D_{Tol} = 1.0E−2).

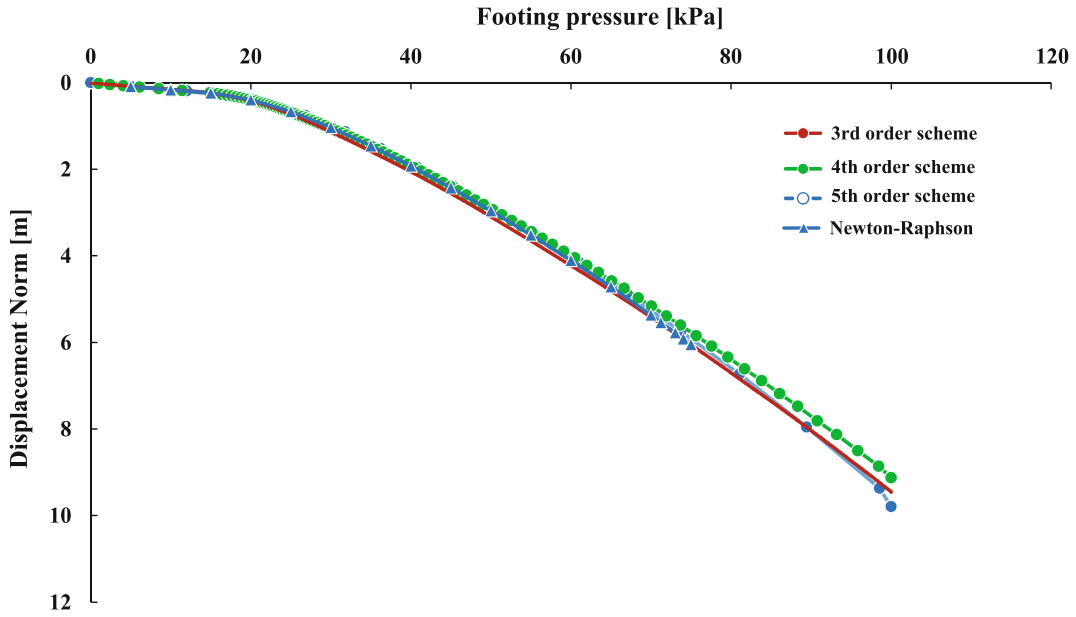

Fig. 10. Numerical results with different schemes and tolerated error (D_{Tol} = 1.0E−3).

where the isotropic pressure $p' = \sigma'_{kk}/3$ and the deviatoric stress invariant $q = \sqrt{\frac{3}{2} s_{ij} s_{ij}}$: $s_{ij} = \sigma'_{ij} - p' \delta_{ij}$, with σ'_{ij} being the effective Cauchy stress tensor and δ_{ij} is the Kronecker delta and i,j = 1,2,3. The model assumes an associated flow rule.

In the case of overconsolidated clay, the initial isotropic preconsolidation pressure is calculated based on the measured pre-overburden pressure *POP* (the maximum vertical effective stress ever experienced by the soil) and Eq. (13) yielding:

$$p_c = POP \times \frac{[M^2(1 + 2K_o^{NC})^2 + 9(K_o^{NC} - 1)^2]}{3M^2(1 + 2K_o^{NC})} \tag{14}$$

where K_o^{NC} is the coefficient of at-rest earth pressure as predicted by MCC model for normally consolidated clay (Brinkgreve and Vermeer, 1992):

$$K_o^{NC} \approx \frac{1.6 - 0.2M}{1 + M} \tag{15}$$

The stress integration in case of MCC model can be carried out using serval standard schemes including the explicit schemes with error control (Sloan, 1987; Sołowski and Gallipoli, 2010) and implicit schemes (Borja and Lee, 1990; Abed, 2008). Thebes code offers both options to integrate MCC model, however in the following examples only the implicit scheme for the local stress integration is used.

8.1. Circular footing on Modified Cam Clay soil

The first example is an analysis of a circular shallow foundation. The 1.0 m radius footing rests on a dry homogeneous slightly over-consolidated clay with *POP* = 20 kPa. This *POP* value, based on Eqs. (14) and (15), leads to an initial isotropic overconsolidation pressure p_c = 18.25 kPa. The unit weight of the clay is assumed to be 17.0 kN/m^3 and the stresses were initialized with at rest soil pressure coefficient K_o = 0.5. Table 8 lists the adopted MCC parameters in this analysis. The footing applies a uniformly distributed stress of 100 kPa directly on the

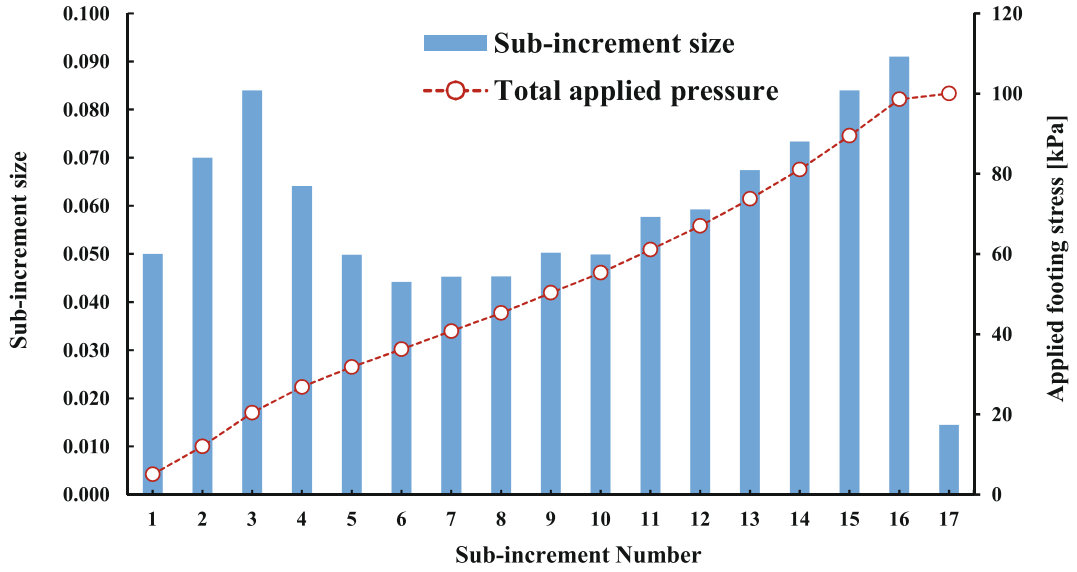

Fig. 11. Sub-increment size variation in case of 5th order scheme with tolerated error (1.0E−3).

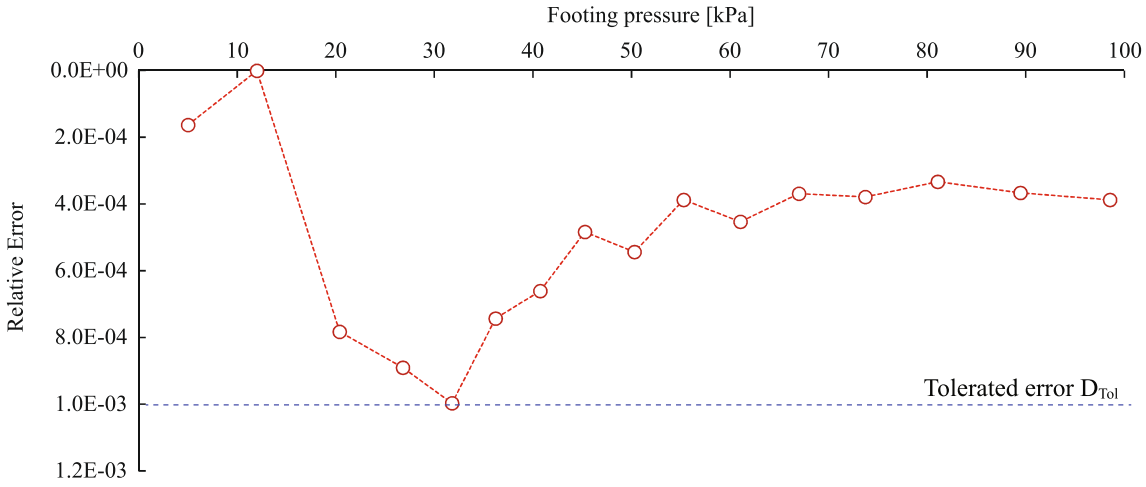

Fig. 12. The relative error in case of 5th order scheme with tolerated error (1.0E−3).

soil in axisymmetric conditions, representing the case of a perfectly flexible footing. The theoretical ultimate bearing capacity of the footing is around 110 kPa (Vesic, 1975). Fig. 4 shows the finite element model and the problem geometry. The mesh consists of 2100 4-noded quadrilateral finite elements with 4 stress integration points per element. Standard mechanical boundary conditions are applied to the problem boundaries with constrained horizontal deformations on the vertical boundaries and fully constrained deformations at the bottom boundary. The numerical analysis is carried out using the finite element code Thebes (Abed and Sołowski, 2017, 2019). Originally, the code used full Newton-Raphson scheme to solve the finite element equations, which will be used here as a reference for comparison purposes. The 100 kPa external stress is applied as one loading increment, and then the automatic scheme divided it into the appropriate sub-increments to satisfy the required accuracy using the desired order of the Runge-Kutta scheme. Automatic sub-incrementation is also used for the Newton-Raphson method to constrain the maximum relative error to I_{TOL} = 1.0E−3. It is worthwhile to mention that the Newton-Raphson sub-incrementation follows similar steps to that in Table 7 with three main differences: 1) The displacements in Step 2 are estimated using Newton-Raphson iterations; 2) The "*Error*" estimation in phase (c) of Step 2 bases on calculated forces instead of displacements where $Error = \|\mathbf{r}_i\|/\mathbf{f}_{ext,i}$. Here $\|\mathbf{r}_i\|$ is the norm of force residual (out-of-balance) and $\mathbf{f}_{ext,i}$ is the applied external load at the current loading level i and 3) the value of β in the phase (d) of Step (2) is estimated as $\beta = \chi\sqrt{\frac{I_{Tol}}{Error}}$.

8.1.1. Discussion of the numerical results

Tables 9–11 report the numerical results of the analyses obtained with a range of displacement error tolerance D_{Tol} and a range of Runge-Kutta schemes of different orders. The Tables give the number of required sub-increments, the number of stages and the relative time with reference to t_{ref} - the time required by the 5th order scheme. Figs. 5–7 graphically show the applied footing pressure versus the Euclidian norm of displacement in the studied domain for the different methods adopted in this study. Each graph includes the results using the Newton-Raphson method for the comparison purposes. For example Fig. 8, Fig. 9 and Fig. 10 illustrate the predictions of 3rd, 4th and 5th order schemes with D_{Tol} = 1.0E−1, 1.0E−2 and 1.0E−3, respectively. The results show that for D_{Tol} = 1.0E−3, all Runge-Kutta schemes approximated successfully the correct solution. However, the number of the required sub-increments to obtain the solution differs significantly from one scheme to another. It is worth noting that in the case of the 5th order scheme, the deviation from the correct solution is relatively small, no matter which tolerance is adopted. For this example, the full Newton-Raphson algorithm diverged around the applied footing stress

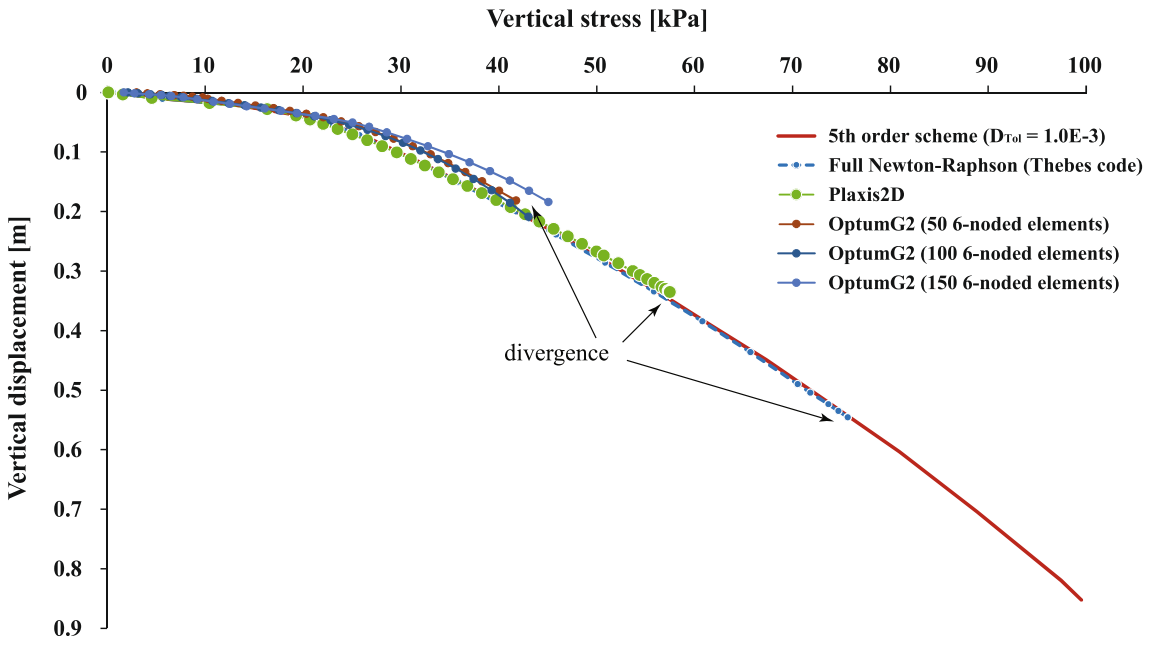

Fig. 13. Predictions of different available numerical codes for the vertical displacements under the centre of the footing.

Table 12
Time and number of sub-increments required by the 5th order scheme with $D_{Tol} = 1.0E-3$ versus time and number of iterations required by full Newton-Raphson method in case of normally consolidated clay.

Scheme	No. successful sub-increments	No. failed sub-increments	Number of required stages/iterations	Calculation Time
5th order	50	0	300	$t_{ref}^{(3)}$
Newton-Raphson	78	0	341	$\approx 1.25t_{ref}^{(3)}$ with divergence at 71 kPa

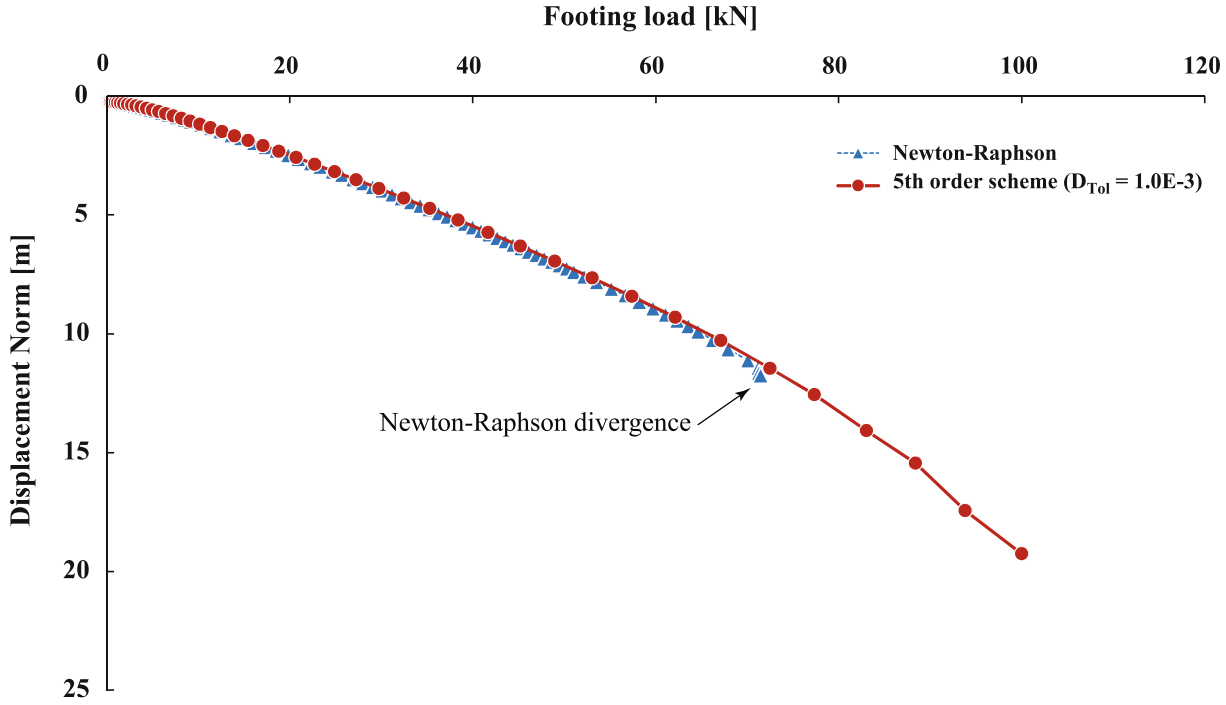

Fig. 14. Numerical results in case of normally consolidated clay.

of 75 kPa illustrating one of the significant merits of the new method, which always succeeded to complete the calculations and provide the results without numerical difficulties. In addition, the 5th order scheme is considerably faster than the Newton-Raphson scheme as it succeeded to compute the results for the full 100 kPa loading in about 25% faster than the time needed by the iterative scheme to reach 75 kPa loading before diverging. That is confirmed by using the approximate assessment of the number of calculations based on count of the total number of calculation stages and iterations. The 5th order scheme completed the calculations with 108 calculation stages versus 131 iterations done by the Newton Raphson scheme before it diverges and triggers continuous unsuccessful attempts to converge by reducing consecutively the load increment size. Additionally, the Newton-Raphson scheme could easily reach 200 iterations if the iterative method would be able to converge and compute the final results (corresponding to full load increment applied).

Table 9, Table 10, Table 11 and Fig. 8, Fig. 9, Fig. 10 illustrate the results of the different schemes but with a similar tolerance. The graphs clearly show the significant differences in the required number of sub-increments (illustrated by dots) in each scheme to reach the prescribed accuracy. It is also clear that the 5th order scheme is the most accurate and requires the lowest number of sub-increments which verifies the theoretical expectations.

Fig. 11 shows the sub-increment sizes with loading for the case of the 5th order scheme and $D_{Tol} = 1.0\text{E}-3$. The graph illustrates the general tendency of the error control scheme to accelerate the increase of the load increment size as long as the load-displacement curve is smooth, and the slope is not changing dramatically (pure elasticity). Once the curve changes in a non-smooth manner (soil becomes plastic as stress reaches the preconsolidation pressure value, which is a non-smooth transition), the load increment size reduces in order to control the error. Once plastic behaviour is present, the load-displacement curve becomes smooth again, the scheme starts accelerating and the step size increases quickly again. Fig. 12 shows that the occurred relative error is kept below $1.0\text{E}-3$ throughout the loading sequence which verifies the capability of the procedure to constrain the error below the prescribed threshold.

8.1.2. Comparison to commercial codes results

Exactly the same problem was modelled using Plaxis2D (Brinkgreve et al., 2016) and OptumG2 (Krabbenhoft et al., 2018), two well-established commercial software for finite element calculations in geotechnical applications. Fig. 13 depicts the results of the calculations in terms of the predicted vertical displacements directly under the centre point of the footing (point A in Fig. 4). Plaxis could not advance further than 60 kPa where the calculations stopped indicating soil collapse. Even after activating the arc-length control option, Plaxis code still diverges at that point. It is worth mentioning that Plaxis uses the modified Newton Raphson iterative scheme to solve the balance equations. A similar observation applies to OptumG2 which shows even earlier divergence around 50 kPa. The full Newton-Raphson method as implemented in Thebes code diverged around 75 kPa while the new Runge-Kutta scheme succeeded to finish the calculations smoothly. These calculations demonstrate the potential of the new method to be competitive if compared to the implicit iterative method.

8.2. Footing on normally consolidated (NC) clay

Table 12 and Fig. 14 report the results of the code predictions for the same footing problem but with an overburden pressure $POP = 1.0$ kPa to model a normally consolidated clay. The analysis involves the Newton-Raphson method and the 5th order explicit scheme. Similar observations to that in the previously discussed case of higher POP of 20 kPa (overconsolidated soil) apply here as well. The analysis run with the 5th order Runge-Kutta scheme completed successfully while the calculations with the iterative Newton Raphson procedure failed under external stress of 71 kPa. In addition, the 5th order scheme required 50 sub-increments and 300 calculations stages to go through the complete loading. The iterative scheme needed 341 iterations to reach 71 kPa before the divergence. The actual relative time needed by the Newton-Raphson method is again about 25% in excess of the time needed by the 5th order scheme to carry out a similar analysis. This example illustrates that for plasticity-dominant-problems, where the iterative procedures need an increasing number of iterations to converge, the 5th order scheme possesses, besides the convergence merits, competency with respect to the analysis efficiency and speed.

9. Conclusions

This paper introduced a new method to solve the load-displacement finite element equations based on Runge-Kutta scheme. The method is implemented with different local truncation errors up to 5th order. Additionally, the procedure includes a routine that automatically controls the load increment size to restrict the load path error within a desired range. The performance of the procedure is verified by solving challenging shallow foundation problems supported by an elastoplastic material. The tackled numerical applications compared the proposed method with other common methods used for the solution, including the Newton-Raphson scheme and the arc-length method. The primary findings are: 1) in comparison to Newton Raphson iterative method, the new explicit method proved to be not only more stable but also competitive with respect to the speed of solution especially when the 5th order Runge-Kutta scheme is employed; 2) the convergence of the proposed scheme is assured even in highly nonlinear problems, while the other schemes fail; 3) the presented procedure is universal, and the proposed method can be extended to even higher-order Runge-Kutta schemes easily. Finally, more research is needed to check whether the benefits of the proposed method are also present in other highly nonlinear problems. For example, those may involve coupled multi-physics where numerical instability and divergence are relatively common, for instance in the case of coupled thermo-hydro-mechanical-chemical (THMC) problems.

CRediT authorship contribution statement

Ayman A. Abed: Methodology, Software, Validation, Writing - original draft, Visualization. **Wojciech T. Sołowski:** Conceptualization, Supervision, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The authors would like to acknowledge gratefully that the presented research has been funded by the Civil Engineering Department's basic research fund at Aalto University. The authors would also like to acknowledge inspiring discussions with Prof. Kristian Krabbenhoft (University of Liverpool), which led to the initiation of this research.

References

Abbo, A.J., Sloan, S.W., 1996. An automatic load stepping algorithm with error control. Int. J. Numer. Methods Eng. 39, 1737–1759.
Abed, A., 2008. Numerical modeling of expansive soil behavior. Instituts für Geotechnik (IGS).
Abed, A., Sołowski, W., 2017. A study on how to couple thermo-hydro-mechanical behaviour of unsaturated soils: Physical equations, numerical implementation and examples. Comput. Geotech. 92, 132–155.
Abed, A.A., Sołowski, W.T., 2019. Applications of the new thermo-hydro-mechanical-

chemical coupled code 'Thebes'. Environ. Geotech. https://doi.org/10.1680/jenge.18.00083.
Avriel, M., 2003. Nonlinear programming: analysis and methods. Courier Corporation.
Bathe, K.-J., 2006. Finite element procedures. Klaus-Jurgen Bathe.
Bischof, C., Carle, A., Corliss, G., Griewank, A., Hovland, P., 1992. ADIFOR–generating derivative codes from Fortran programs. Sci. Program 1, 11–29.
Bischof, C., Khademi, P., Mauer, A., Carle, A., 1996. ADIFOR 2.0: Automatic differentiation of Fortran 77 programs. IEEE Comput. Sci. Eng. 3, 18–32.
Borja, R.I., Lee, S.R., 1990. Cam-clay plasticity, part 1: implicit integration of elasto-plastic constitutive relations. Comput. ApplMechEng. 78, 49–72.
Brinkgreve, R., Kumarswamy, S., Swolfs, W., 2016. PLAXIS 2016. Plaxis bv, Delft.
Brinkgreve, R.B.J., Vermeer, A., 1992. On the use of Cam-Clay models. Symp. Numer. Models Geomech. Balkema Rotterdam Neth.
Broyden, C.G., 1965. A class of methods for solving nonlinear simultaneous equations. Math. Comput. 19, 577–593.
Crisfield, M.A., 1983. An arc-length method including line searches and accelerations. Int. J. Numer. Methods Eng. 19, 1269–1289.
Gens, A., Potts, D.M., 1988. Critical state models in computational geomechanics. Eng. Comput. 5, 178–197.
Griewank, A., Walther, A., 2008. Evaluating derivatives: principles and techniques of algorithmic differentiation, vol. 105. Siam.
Gustafsson, K., 1992. Control of error and convergence in ODE solvers.
Krabbenhoft, K., Lymain, A., Krabbenhoft, J., 2018. OptumG2. Optum Comput. Eng.
Laitinen, M., 2013. Numerrin 4.0 Manual. Numerola Oy.
Lee, H.J., Schiesser, W.E., 2003. Ordinary and partial differential equation routines in C, C++, Fortran, Java, Maple, and Matlab. Chapman and Hall/CRC.
Matthies, H., Strang, G., 1979. The solution of nonlinear finite element equations. Int. J. Numer. Methods Eng. 14, 1613–1626.
Mitusch, S.K., 2018. An Algorithmic Differentiation Tool for FEniCS.
Potts, D.M., Ganendra, D., 1992. A comparison of solution strategies for non-linear finite element analysis of geotechnical problems. Proc. 3rd Int. Conf. Comput. Plast. Barc. 803–814.
Rothe, S., Hartmann, S., 2015. Automatic differentiation for stress and consistent tangent computation. Arch. Appl. Mech. 85, 1103–1125.
Schofield, A., Wroth, P., 1968. Critical state soil mechanics.
Sheng, D., Sloan, S.W., 2001. Load stepping schemes for critical state models. Int. J. Numer. Methods Eng. 50, 67–93.
Sloan, S.W., 1987. Substepping schemes for the numerical integration of elastoplastic stress–strain relations. Int. J. Numer. Methods Eng. 24, 893–911.
Sloan, S.W., Abbo, A.J., Sheng, D., 2001. Refined explicit integration of elastoplastic models with automatic error control. EngComput. 18, 121–194.
Smith, I.M., Griffiths, D.V., Margetts, L., 2013. Programming the finite element method. John Wiley & Sons.
Sołowski, W.T., Gallipoli, D., 2010. Explicit stress integration with error control for the Barcelona Basic Model: Part I: Algorithms formulations. ComputGeotech. 37, 59–67.
Tijskens, E., Roose, D., Ramon, H., De Baerdemaeker, J., 2002. Automatic differentiation for solving nonlinear partial differential equations: an efficient operator overloading approach. Numer. Algorithms 30, 259–301.
Vesic, A.S., 1975. Bearing capacity of shallow foundations. Found Eng. Handb.
Wood, D.M., 1990. Soil behaviour and critical state soil mechanics. Cambridge University Press.
Zwicke, F., Knechtges, P., Behr, M., Elgeti, S., 2016. Automatic implementation of material laws: Jacobian calculation in a finite element code with TAPENADE. Comput. Math. Appl. 72, 2808–2822.